GEORGIA INSTITUTE OF TECHNOLOGY

School of Electrical and Computer Engineering

ECE-6276-A DSP Hardware System Design (Fall 2018)

Lab 5: FIR Filter Design

Assigned Date : Wednesday, September 26, 2018

Due Date : 23:55, Thursday, October 4, 2018

---

1 Introduction

A digital FIR Filter is basically a set of MAC operations specified in an order depending upon on the nature of the filter. We will be designing an 8-tap filter in this lab. The filter equation is

$$y[n] = \sum_{i=0}^{7} h[i]x[n-i]$$

Here, the coefficients $h[i]$ have real and imaginary parts, both of which are 8-bit signed fix-point numbers and are hard-coded in the design. Also, note that the multiplication and addition are signed operations. The template for the FIR filter is provided in `filter_8.vhd` in `./src` directory.

The explanations of the input and output ports are given below.

- `clk` => Clock signal. We use rising edge for flops.
- `rst` => Reset signal. Note that the sequential elements that are used in your design should have **synchronous** active high reset. i.e. if the reset is pulled high, the flops should be reset at next rising edge of clk.
- `in_valid` => `in_valid` is the qualifier for the data. When `in_valid = 1`, the data at the same cycle is valid.
- `next_in` => `next_in` is generated by the design and tells the testbench that, if `next_in = 1`, the testbench can feed next data set, otherwise the testbench has to hold the current valid data.
- `data_in` => The input data port from which data is passed to the design. Note that the real 8-bit fix-point data is passed to the design serially.
- `data_real_out and data_imag_out` => Outputs from the design. After the complex operations, the real data which is passed into the input is transformed into real and imaginary components.

- `out_valid => valid signal` for the output. When `out_valid = 1`, the outputs `data_real_out` and `data_imag_out` are valid. This information is useful to the testbench which captures the data only when output is valid.
- `next_out => next_out` is generated by the testbench and tells the design that, if `next_out = 1`, the design can generate next output data set, otherwise the design has to hold the current valid data.

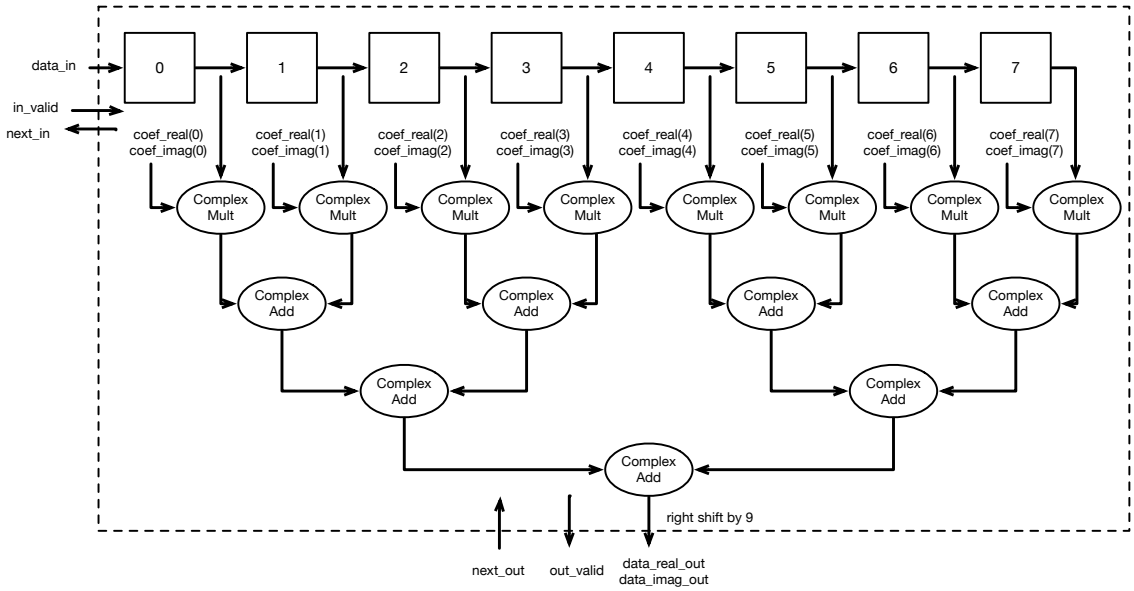

Figure 1 8-tap FIR filter with real data and complex coefficients

This system can be implemented in numerous ways with different latencies, area and power. You can use any architecture as long as you can clearly draw the block diagram of your architecture in the report.

There is an input file `input_seq.txt` from which the data is read into the testbench, as well as an input file `input_out_ready.txt` from which the `next_out` is to be assigned. The testbench dumps out an output file. Compare this output with the reference output file `output_ref.txt` provided with the assignment. The output will be the same for any architecture. The grading will take into consideration the functional correctness, utilization (area), timing and the power of the design. Three re-submissions are allowed for this lab. The resources information is obtained from `filter_8_utilization_placed.rpt`. The timing information is obtained from `filter_8_timing_summary_routed.rpt`. The power information is obtained from `filter_8_power_routed.rpt`.

Note that our design may be very bad for area but is quick. So the testbench gives a data sample every cycle, if possible. In your design, you can choose to use a memory/buffer which stores all these samples and will parse it to main logic at a speed which is decided by the latency of your architecture. This additional memory overhead will not be considered for grading.

2 Instructions

1. Draw the block diagram of the architecutre you will be using. (If you are using the same architecture as the figure above, then you don't need to draw it)
2. Complete your design in VHDL. You can use multiple design files this time, but the top file should be `filter_8.vhd`.
3. Run the simulation for `filter_8.vhd` using the provided testbench `tb_filter_8.vhd` to match your output file `output.txt` with the reference `output_ref.txt`.
4. Run the synthesis and implementation of your design `filter_8.vhd` in Vivado using the provided constraint file `constraints.xdc`.

3 Deliverables

1. Create a PDF report containing:
   - Block diagram of your design (If you are using the same architecture as the figure above, then you don't need to draw it)
   - Tables of
     (a) Resources (Only "Slice Logic", "IO and GT Specific", "Primitives") from `filter_8_utilization_placed.rpt`
     (b) Power (Dynamic and Static) from `filter_8_power_routed.rpt`
     (c) Worst Negative Slack ("Design Timing Summary") from `filter_8_timing_summary_routed.rpt`
   - (15% of grade) Answer the question: Why do you think we need to do right shift operation? And why do we do it at the end? (Answer should not exceed 6 sentences)

   The name of the PDF should be of the form `lab5_firstname_lastname.pdf` , e.g. `lab5_george_burdell.pdf` .
2. (75% of grade for this bullet and the next) The completed design file `filter_8.vhd` . **(No latch generated; synchronous reset, sensitivity signal list correct)**
3. Other modules (if applicable)
4. The simulated output file `output.txt` in the "run" folder.
5. (10% of grade) The implemented area (utilization), power and timing reports, namely,

   `filter_8_utilization_placed.rpt`

   `filter_8_power_routed.rpt`

   `filter_8_timing_summary_routed.rpt`

Move all of these files into a folder called `lab5_firstname_lastname_gtID` . Zip the folder and upload the archive `lab5_firstname_lastname_gtID.zip` on T-Square (eg. `lab5_george_burdell_123456789.zip`). The first name and last name should be all in lower case. **Please strictly follow this naming convention. Otherwise my script will not work and you might get points deduction.**

Note: Late submissions are not accepted. In case of extraordinary circumstances, written permission must be obtained from Dr. Madisetti.